Dynamic L4-L7 Service Insertion with Cisco ACI and A10 Thunder ADC

Table of Contents

Executive Summary 3
Document Scope 3
Audience 3
Business Challenge 3
Business Solution 4
Solution Architecture 5
- Cisco Application Policy Infrastructure Controller (APIC) 6
- Cisco Nexus 9000 Series Switches 7
- A10 Networks Thunder ADC 8
- A10 Harmony Architecture, Programmability and SDN Integration 8
- Cisco ACI Fabric with A10 Thunder ADC Devices 8

A10 Thunder ADC Integration with Cisco ACI 9
ACI Design with A10 Thunder ADC 10
L4-L7 Service Graph Instances 12
Adding L4-L7 Devices to the APIC 16
Verifying L4-L7 Service Graph Configuration 18
Reference Architecture Components 21
Summary 21
About A10 Networks 21

List of Figures

Figure 1: Cisco ACI and A10 Thunder ADC 4
Figure 2: Cisco ACI architecture 5
Figure 3: L4-L7 service graph 6
Figure 4: Cisco APIC showing A10 device package 7
Figure 5: Leaf and spine architecture using Nexus 9000 Series Switches (source: Cisco) 7
Figure 6: Cisco ACI architecture with A10 Thunder ADC appliances 9
Figure 7: Cisco ACI integration – A10 device package 9
Figure 8: ACI design with A10 Thunder ADC 10
Figure 9: Two-tier application network profile 10
Figure 10: Application profile "webx" with four EPGs 11
Figure 11: Service graph creation example 12
Figure 12: APIC rendering service graph to deploy application profile 13
Figure 13: Service graph instance example 13
Figure 14: Service graph showing L4-L7 function node configuration 14
Figure 15: Associating service graph with a security contract 15
Figure 16: APIC GUI showing Deployed Service Graph Instances 15
Figure 17: L4-L7 device selection policy 16
Figure 18: Adding L4-L7 device to tenant 17
Figure 19: L4-L7 device status in APIC GUI 17
Figure 20: Creating a logical device context 18
Figure 21: APIC GUI deployed graph configuration information 18

Executive Summary

Traditional data center infrastructures are inflexible, difficult to change and don't scale well, making it hard to meet today's demanding data center needs in a cost-effective manner. Teams that build and manage data center infrastructure traditionally operate in their own silos, with no common operational model between application, network and security. As a result, new application rollout takes anywhere between a few weeks to months of effort often due to poor coordination between the different teams managing data center infrastructure.

One way to solve this problem is by making use of programmable application services to build new data center infrastructure, which will deliver consistent policy enforcement across all applications in a timely manner. Cisco Application Centric Infrastructure (ACI), together with A10 Networks® Thunder® ADC line of Application Delivery Controllers, allows organizations to respond rapidly to business demands by enabling automatic provisioning of dynamic L4-L7 application networking and security services. This integrated solution delivers automated policy enforcement for business applications in an application-centric, programmable approach by combining applications, networking, security and the underlying infrastructure.

Document Scope

The purpose of this reference architecture is to serve as a design guide for the dynamic L4-L7 service insertion solution architecture for A10 Thunder ADC L4-L7 services with Cisco ACI. This joint solution consists of Cisco Nexus 9000 switches, Cisco Application Policy Infrastructure Controller (APIC), and A10 Thunder ADC physical and virtual appliances. This reference architecture explains how to auto provision and deploy A10 Thunder ADC devices in a Cisco ACI design using Cisco APIC with the A10 device package. Examples of A10 Thunder ADC L4-L7 services deployment using Cisco APIC GUI and XML post scripts are also included in this document.

Audience

Cloud, networking and data center architects who are responsible for planning, designing and implementing network architecture for modern data centers are the primary target audience for this document. The document should also be useful to system engineers and professional services specialists who are responsible for deploying and managing public or private cloud data center infrastructure built using Cisco ACI. Readers should have some degree of familiarity with Cisco ACI and A10 Thunder ADC.

Business Challenge

Traditional data center infrastructures are inflexible and don't scale well beyond a certain point. This makes it difficult to meet business agility and scaling needs in a cost-effectively manner. Applications, network and security teams who build and manage data center infrastructure traditionally operate in their respective silos, which means that there is no common operational model between them. Provisioning applications and addressing their changing network services needs are time-consuming and require significant process overhead. Manual service provisioning creates operational challenges to scale up and quickly deploy resources for a broad range of applications while ensuring application availability, performance and security. As a result, the ability of data center administrators to react quickly to business demands is constrained.

Business agility comes from application agility – the ability to roll out and provision new applications in hours instead of days, weeks and months. Rolling out new applications typically requires orchestration of multiple resources at the compute, network and storage services level in a coordinated fashion inside the Infrastructure-as-a-Service (IaaS) layer and/or orchestration of applications being hosted in the cloud.

Cloud IaaS delivers vastly greater business agility, service provisioning times and economics. However, Cloud IaaS requires that the underlying infrastructure is automated and scalable to build an on-demand delivery model providing dynamic and consistent services in a shared, multi-tenant environment. To build this model, it is important to put applications, networking, security and the underlying infrastructure in a common operational model that is optimized at solving business problems. This application-centric model should include application networking services with other data center infrastructure to deliver dynamic L4-L7 services and automated policy enforcement.

Business Solution

The new data center design has to be business-focused, with emphasis on applications and programmability to provide agility and simplicity in data center infrastructure. Software Defined Networking (SDN) technologies look promising as they bring a similar degree of agility to networks that abstraction, virtualization and orchestration have brought to server infrastructure. Multiple vendors are working on different approaches to offer SDN products and services: While some vendors are creating SDN-enabled switches, routers and network equipment, others are taking a more holistic software approach that would allow the network to be virtualized through central controllers.

Figure 1 shows Cisco ACI powered by Cisco Nexus 9000 Series Switches, Cisco APIC and A10 Thunder ADC appliances. In this design, Cisco APIC acts as the centralized policy controller that manages and controls the infrastructure. Cisco Nexus 9000 Series Switches provide the L2-L3 fabric for ACI. And A10 Thunder ADC devices provide L4-L7 services. IT organizations can now leverage this joint solution to transform their data center to an application-centric model with significant increase in provisioning speed, business agility and reduced costs.

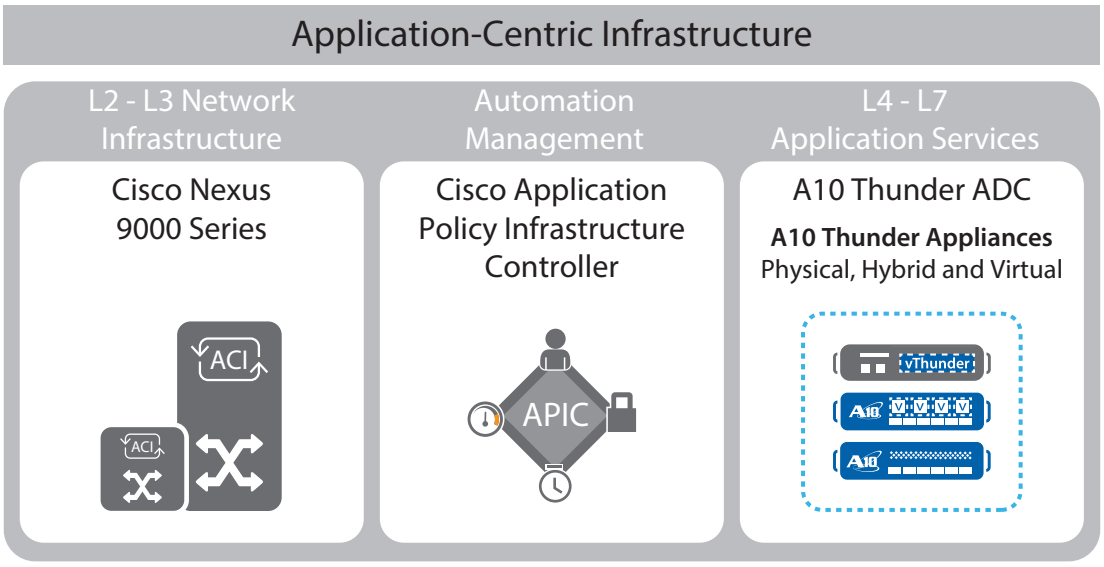

Figure 1: Cisco ACI and A10 Thunder ADC

The Cisco ACI fabric is designed as an application-centric intelligent network, where the Cisco APIC policy model is defined from the top down as a policy enforcement engine focused on the application itself and abstracting the networking functionality underneath. The A10 integration with Cisco ACI ensures that network segmentation and security policies are enforced consistently whenever a new application is deployed in the network. IT organizations can leverage the comprehensive joint solution to transform their data center as applications and services are created with significant increase in provisioning speed, business agility and reduced costs.

Evolving applications require that IT organizations ensure a seamless infrastructure for application deployment and on-demand consumption. It is critical to build modern data center infrastructures that deliver consistent application and security services in an elastic and cost-effective environment. The biggest challenge with current networking technologies is the tight coupling between networking protocols and features, forwarding and security policy. As a result, any change in network policy is very likely to adversely affect traffic forwarding and disrupt flow.

In the new approach to building data center infrastructures, there is a focus on combining applications, networking, security and the underlying infrastructure in a common operational model optimized for solving business problems. Programmability is the glue that is the key to delivering consistent policy enforcement across all applications, whether they are custom or off-the-shelf. It is critically important to have seamless integration of application networking services with other data center infrastructure to deliver dynamic L4-L7 services and automated policy enforcement for SLAs, as well as security compliance in a shared, multi-tenant environment.

Solution Architecture

The solution architecture for a next-generation data center consists of A10 Thunder ADC and Cisco ACI working together to build an application-centric networking infrastructure. Cisco ACI provides centralized, application-driven policy automation and management for all layers of the networking infrastructure. Thunder ADC integrates with the Cisco ACI open ecosystem model and provides dynamic L4-L7 application networking services that can be automated and managed by Cisco ACI APIC. This architecture offers maximum flexibility and accelerates application deployments by simplifying and optimizing the deployment process for new applications, both traditional and custom. This automated approach is well suited for DevOps environments, which rely heavily on programmability of application services, networking and security policies. Multi-tenancy is a key aspect of the ACI fabric, which enables managed service providers to deploy applications in a secure and isolated environment, without bothering about the physical infrastructure. The ACI fabric is comprised of the following key components that build the underlying network infrastructure to support applications.

1. Cisco APIC – provides centralized policy management and control
2. Cisco Nexus 9000 Series Switches – provide leaf and spine fabric infrastructure
3. L4-L7 application and security service devices through open ecosystem

The open ecosystem model of ACI allows it to integrate with stateless hardware and software systems where the APIC programs, manages and configures the physical and virtual compute, network and storage infrastructure. The APIC pushes configuration to the switches, routers, firewalls and load balancers through RESTful API calls that allow the data center administrator to replace any failed hardware or system with a similar one without losing any state information or configuration (the configuration resides centrally with the APIC).

Figure 2 shows a high level overview of the Cisco ACI architecture with the APIC managing and interacting with different elements of the data center. This application-centric model can be used by DevOps teams, for example, who could use Cisco APIC as the centralized controller to programmatically configure network infrastructure, application services and security policies through device-specific REST API calls – all at the same time. This approach can greatly reduce the time it takes to plan, implement and provision networking infrastructure to deploy new applications in the data center.

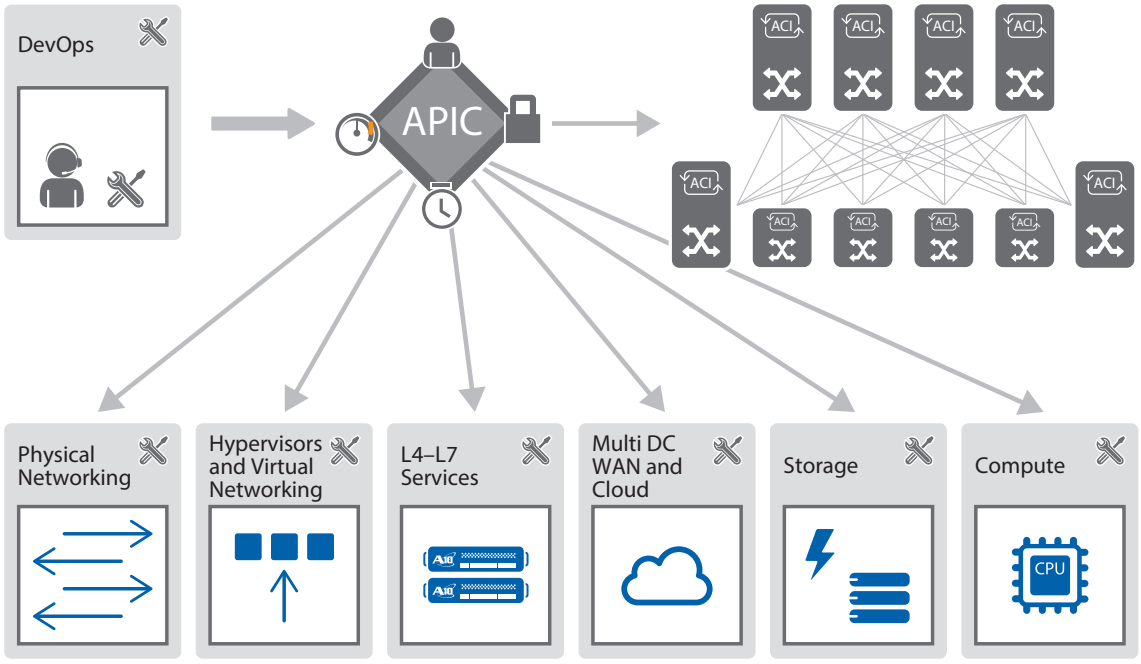

Figure 2: Cisco ACI architecture

Cisco Application Policy Infrastructure Controller (APIC)

Within Cisco ACI, the Application Policy Infrastructure Controller (APIC) is the creation, repository and enforcement point for application policies, which can be set based on application-specific network requirements. The Cisco APIC policy model provides automated L4-L7 service insertion capabilities through a concept called service graph. The service graph shown in Figure 3 represents the ordered set of service nodes for a given application profile. The service graph identifies the set of network or service functions (firewall, server load balancing, SSL offload, etc.) that are required by a policy for an application, and each function is represented as a node in the graph.

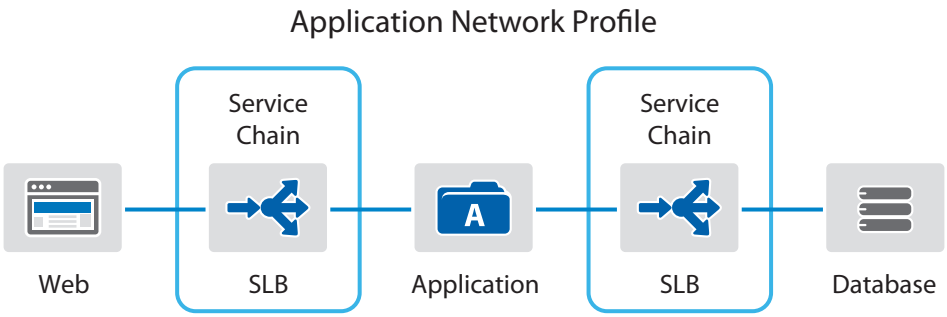

Figure 3: L4-L7 service graph

The APIC automatically provisions and configures the L4-L7 service nodes from the available L4-L7 device pool and stitches them together to create the data path forwarding according to the service graph. The APIC also automatically configures the underlying network infrastructure for data path forwarding according to the needs of the service nodes as specified in the service graph. A service graph typically represents two or more tiers of an application, with the appropriate service function inserted as service nodes in the graph. The service graph is a template that is rendered using physical resources in the data center by APIC.

APIC renders the service graph by allocating a preconfigured pool of L4-L7 devices like load balancers and firewalls to the service nodes. The L4-L7 devices are inserted in the data forwarding path based on the device selection criteria for the tenant. Cisco APIC communicates with the L4-L7 application and security devices using a preinstalled device package that helps it configure and monitor those devices. Figure 4 shows a device package from A10 Networks that has been added to the Cisco APIC. The device package in this particular case enables APIC to configure and deploy L4-L7 ADC services using A10 Thunder ADC devices.

Cisco APIC appliances are deployed as a cluster of fully redundant and load-balanced controllers with a minimum recommendation of three. APIC provides multiple options to configure service graphs, some of which are:

- APIC GUI interface available from a standard web browser
- REST API calls with XML or JSON formatted payloads sent to the APIC
- CLI option to navigate and configure object model from the APIC CLI interface
- Python scripts using Cisco ACI libraries to control and configure APIC

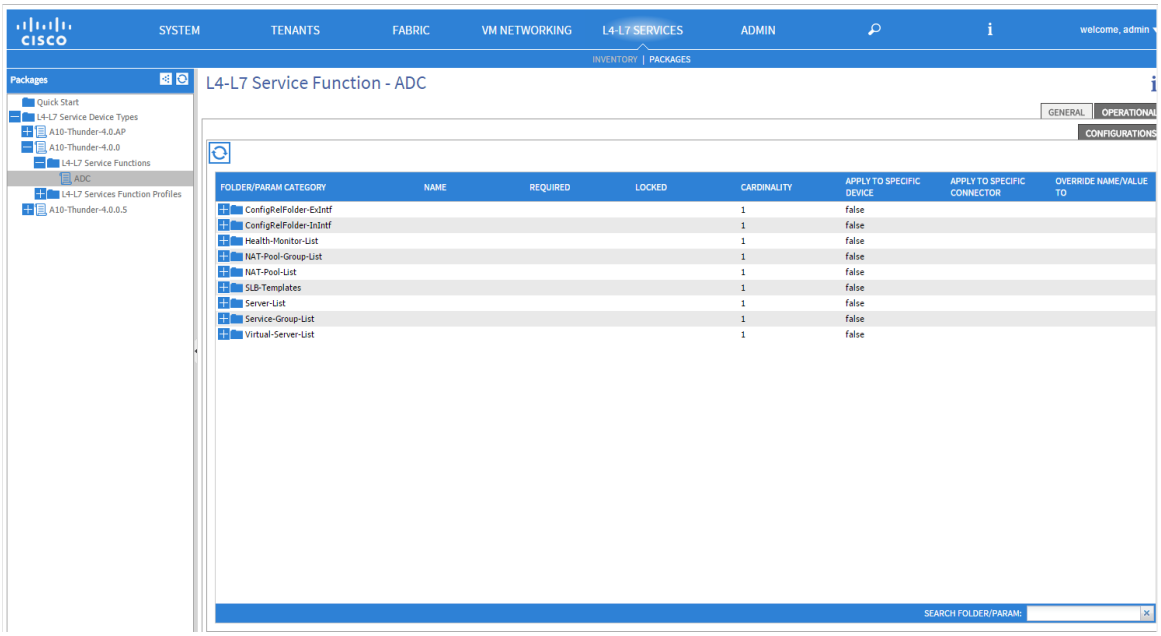

Figure 4: Cisco APIC showing A10 device package

Cisco Nexus 9000 Series Switches

Cisco Nexus 9000 Series Switches are the latest generation switches from Cisco available in fixed and modular 1/10/40 Gigabit Ethernet configuration. The Nexus 9000 switches can operate in two modes:

1. Traditional mode when running the Cisco NX-OS software and can be managed as a standalone switch
2. ACI mode when running the Cisco ACI version of software and will be policy managed through a centralized APIC

In this design, the Nexus 9000 switches in ACI mode are connected in a leaf and spine architecture as shown in Figure 5. In this two-tier architecture, the Nexus 9300 switches are deployed as the leaf switches with 10GbE ports available for attaching endpoints, which could be servers, storage, APIC, security and application service devices such as A10 Thunder ADC and other non ACI switches and routers. The Nexus 9500 switches are deployed as spine switches with 40GbE ports that operate as ACI fabric ports and are used to provide fabric connectivity to the leaf switches. Both the leaf and spine switches can be controlled and managed through Cisco APIC. The physical A10 Thunder ADC devices can be connected to any leaf switch as physical connectivity doesn't really matter in the ACI architecture. The A10 Networks vThunder® ADC line of virtual devices can be running inside a hypervisor on any physical server that is attached to the leaf switch as well.

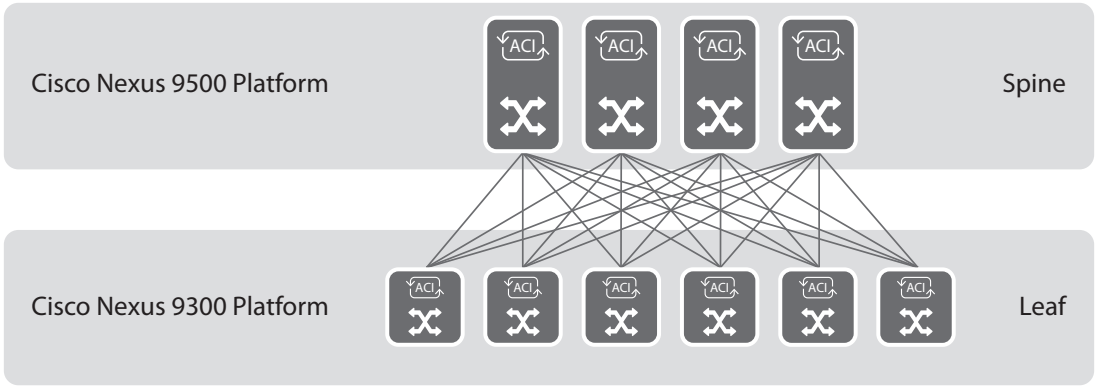

Figure 5: Leaf and spine architecture using Nexus 9000 Series Switches (source: Cisco)

A10 Networks Thunder ADC

A10 Thunder ADC appliances are the next-generation application delivery controllers running A10's Advanced Core Operation System (ACOS®) that delivers high performance and scalability suited for cloud scale environments. The comprehensive features, rich services and flexible architecture of ACOS powered by A10 Harmony allow A10 devices to maximize utilization of CPU resources, and accelerate service integration and manageability via open and standards-based programmability. A10 Harmony programmatically exposes rich features for cloud and network functions virtualization (NFV) environments and ensures flexibility and simplified operations.

The A10 Thunder ADC appliances are available in physical, virtual or hybrid form factor offering flexible options for deployment. ADC features and configurations are identical and transferable across all form factors so that customers can choose any form factor that meets the needs of their environment.

Thunder ADC hardware appliances: The physical A10 Thunder appliances come with 1/10/40/100 Gigabit Ethernet port options and can be connected directly to the Cisco Nexus 9000 series leaf switches. These hardware appliances offer high performance and can scale above 150 Gbps of ADC/server load balancing throughput.

vThunder ADC virtual appliances: The virtual machine version of Thunder appliances, vThunder, runs on almost all industry leading hypervisors and offers maximum flexibility when it comes to deployment choice. The vThunder virtual machine (VM) is supported on VMware, KVM, Citrix Xen and Microsoft Hyper-V hypervisor platforms.

Thunder HVA hybrid appliances: A10 Networks Thunder 3030S HVA appliances offer multiple independent virtual ADC VMs running on a hardware device that offers multi-tenancy with strong tenant isolation, data separation and per VM policy management. The HVA appliances provide hardware acceleration and also enable tenants to orchestrate new instances on demand according to the predefined policies.

A10 Harmony Architecture, Programmability and SDN Integration

A10 Harmony provides the next-generation architecture for A10 Thunder ADC platforms, which lay the foundation of rapid services integration for enterprise, cloud and service provider networks. With A10 Harmony, developers can integrate intelligent application networking services more easily with high-performance A10 Thunder Series product lines. A10 Harmony accelerates service integration and manageability through the open standards-based A10 Networks aXAPI® REST-based API and SDK that enable effective provisioning, telemetry for rich analytics and comprehensive security for application networking and security development. The A10 device package for Cisco ACI is built on the foundation of A10 Harmony architecture and leverages aXAPI SDK and RESTful APIs to provide server load balancing and application delivery services for Cisco ACI using high-performance cloud, virtual or physical Thunder ADC appliances.

Cisco ACI Fabric with A10 Thunder ADC Devices

The Cisco ACI data center design has three or more APICs attached to the Nexus 9000 series leaf switches that function as the centralized policy controller. The ACI fabric consists of Cisco Nexus 9000 Series Switches serving the leaf and spine switch functions. Cisco APIC serves as the single point of automation and fabric element management in both physical and virtual environments. The APIC communicates with other elements in the fabric via a plugin, also known as device package. This design approach is agnostic to physical layout and it doesn't matter how the elements of the data center are physically connected to the ACI fabric built using Nexus 9000 switches. The physical servers, storage, appliances and other vendor equipment can be placed anywhere in the fabric. The virtualized elements could be running on any virtualized server host that is physically connected to the fabric. The physical servers and appliances connected to the fabric could also run multiple instances of virtual machines. The A10 Thunder ADC physical devices can be attached to the leaf switches directly as shown in Figure 6. The vThunder ADC virtual devices can run on any virtualized server that is directly attached to the leaf switch as well. Figure 6 shows a physical Cisco ACI fabric with various elements of the data center attached to it.

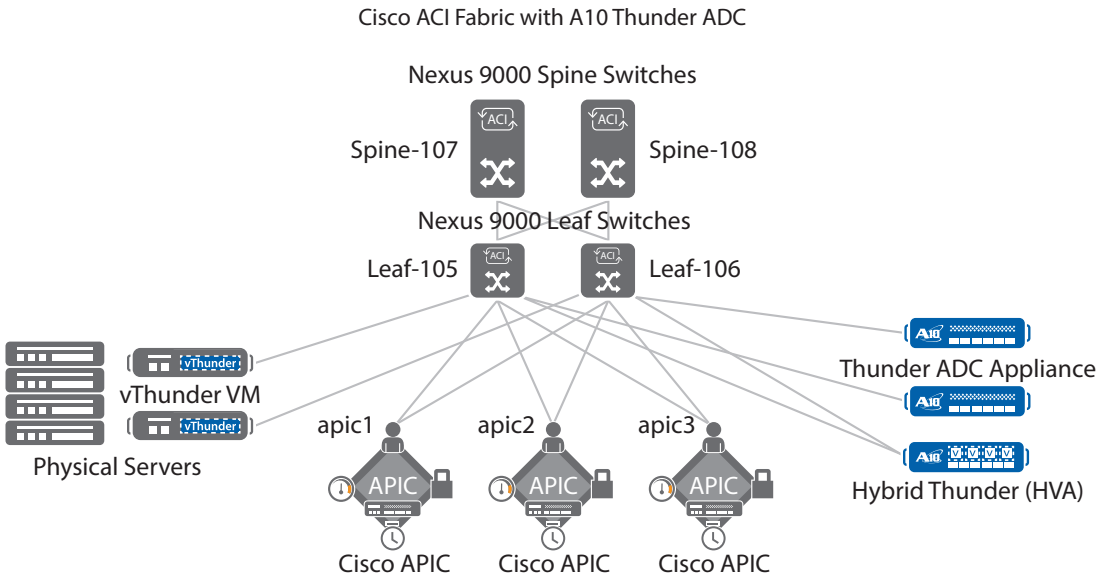

Figure 6: Cisco ACI architecture with A10 Thunder ADC appliances

A10 Thunder ADC Integration with Cisco ACI

Cisco ACI APIC enables interoperability between a Cisco ACI environment and management, orchestration, virtualization and L4-L7 services. With A10 Harmony, developers can integrate intelligent application networking services more easily with high-performance A10 Thunder Series product lines.

Cisco APIC requires a plugin, referred to as a device package, to communicate with the Thunder ADC physical or virtual devices. The A10 device package contains the XML configuration file with A10 device specifications and Python scripts that allow the APIC to interface with A10 devices and deploy services. Figure 7 shows the A10 device package integrated with Cisco ACI architecture.

This integration between A10 and Cisco ACI ensures that network segmentation and security policies are enforced consistently whenever a new application is deployed in the network. Cisco APIC can now manage an A10 device and configure L4-L7 ADC services using RESTful API (aXAPI) commands as the device package provides the translation from APIC callouts to A10 aXAPI commands. The integration leverages A10's ACOS which is completely API driven, enabling delivery of rich services programmatically.

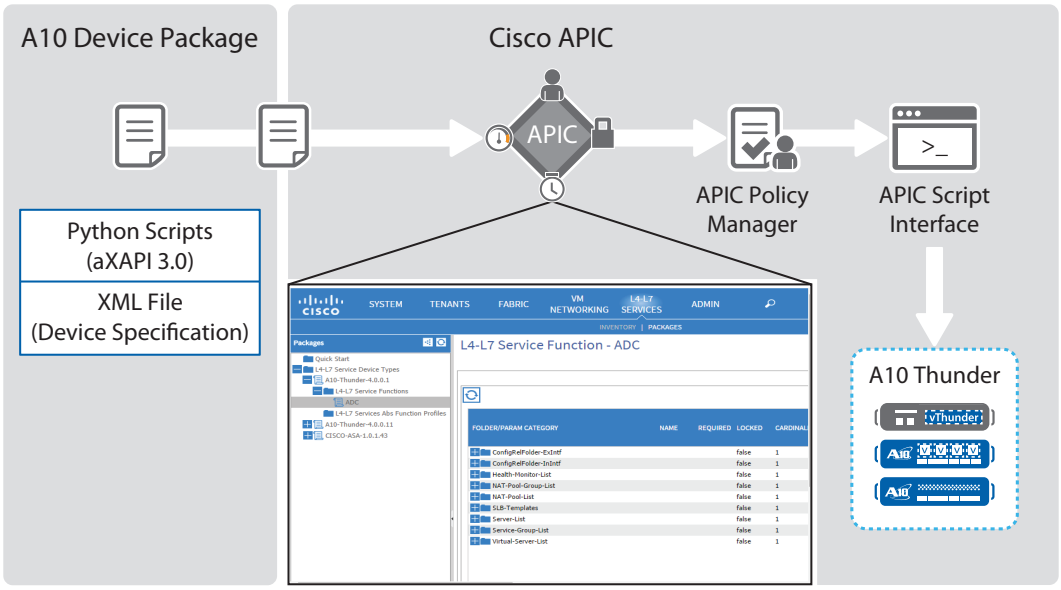

Figure 7: Cisco ACI integration – A10 device package

ACI Design with A10 Thunder ADC

The Cisco ACI fabric consists of L2 and L3 infrastructure built using Nexus 9000 switches in a leaf and spine architecture. The L4-L7 appliances such as load balancers and firewalls can be attached to any ports on the leaf switches. The entire ACI fabric acts like a unified big switch that can be configured and managed through a centralized policy controller (aka APIC). Endpoints, which can be attached to the leaf switches, include virtualized and bare-metal servers, storage devices, other switches and routers. The application services can be hosted on either physical bare-metal servers or virtualized servers that are connected to the leaf switches. Figure 8 shows the ACI powered data center with A10 devices and compute, storage and other infrastructure components.

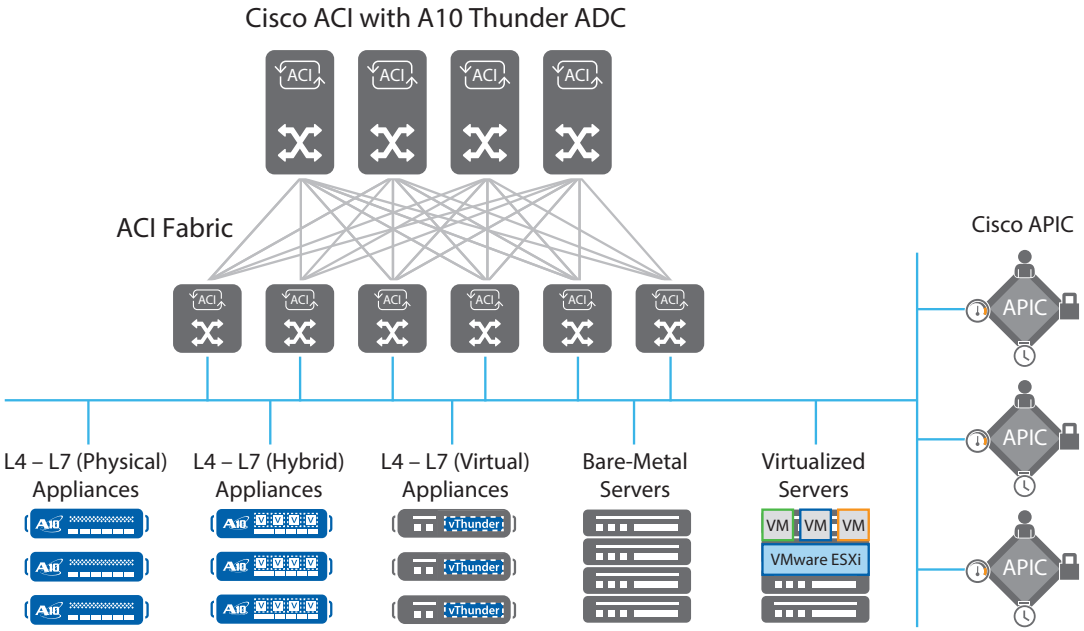

Figure 8: ACI design with A10 Thunder ADC

In the Cisco ACI model, applications are defined logically using constructs that are application centric, rather than network centric. The grouping of resources or entities is done based on logical segmentation; it is no longer based on physical or virtual resources. For example, a group of physical and virtual servers could be grouped together if they are essentially providing the same type of application services. The ACI model supports multi-tenancy where the logical entity supports grouping of applications, network connectivity and security policy enforcement in a contained logical entity. Figure 9 shows a simple example of a two-tier application network that can be hosted on the Cisco ACI fabric in an automated and programmable manner. This example can be extended to a three-tier application as shown in Figure 3 or any multi-tier application hosted on the ACI fabric.

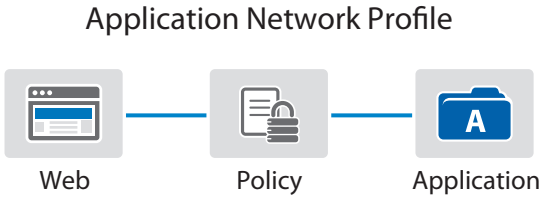

Figure 9: Two-tier application network profile

The web tier consists of a front end tier that serves the web consumers or users. The application tier hosts the servers that provide the content to the web users. The web and application tiers here represent two Endpoint Groups (EPGs). Now we will discuss some of the core concepts of ACI. An understanding of these concepts is essential in the subsequent sections that discuss application profile configuration and the L4-L7 service graph used to provision and deploy applications on top of an ACI fabric.

The following are some of the key ACI concepts to understand how ACI applies an application-centric approach to provision the network infrastructure and support the applications.

Leaf Switches: Leaf switches provide 10GbE and 40GbE connectivity for endpoints and non ACI switches and routers. Cisco APIC is also attached to the leaf switches. These switches are policy controlled by APIC.

Spine Switches: Spine switches provide high-density 40GbE connectivity between leaf switches. These switches form the backbone of the ACI fabric providing one hop connectivity to the leaf switches.

Tenant: Tenant is a logical container for an application policy. The container object allows segregation of resources from a policy perspective. The key objects contained in the tenant are application profiles, bridge domains, Endpoint Groups (EPGs), and security contracts and service graphs.

Application Profile: The application profile models application requirements and contains EPGs. It acts as a logical container for grouping EPGs that are needed for an application. An application profile may contain one or more application EPGs.

Bridge Domain: This represents an L2 forwarding construct within the fabric. A bridge domain can have one or more subnets associated with it, and can be tied with one or more EPGs.

Endpoint Group (EPG): EPG is the most important object in the policy model. It is a logical representation of a collection of endpoints such as servers, virtual machines, network interface cards (NICs), clients on the Internet, etc. In the ACI model, the security policies are applied through the notion of an EPG (and not directly on an endpoint).

Contracts: The security contract object dictates how endpoints in one EPG communicate with those in another EPG. There is no communication allowed between EPGs by default; for example, a device in EPG1 can't talk to a device in EPG2 if there is no security contract between the two EPGs that allow communication. Devices within the same EPG can talk to each other.

Context: The context represents a unique L3 forwarding domain. It can contain multiple bridge domains, and outside network object policies control connectivity to outside of the tenant.

Figure 10 shows a tenant with an application profile named "webx" that has four EPGs attached to it.

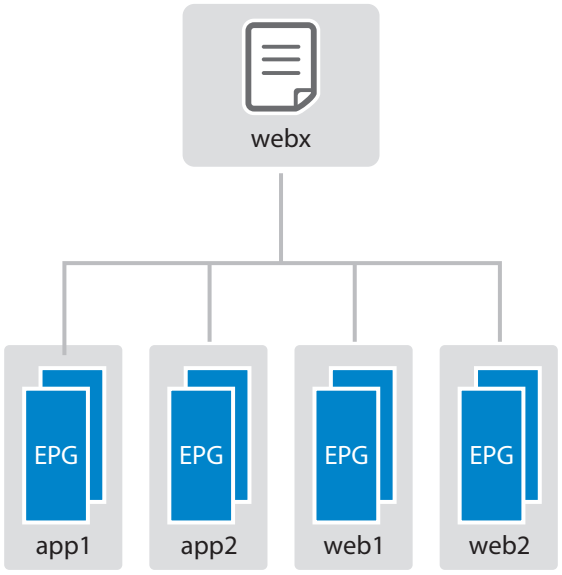

Figure 10: Application profile "webx" with four EPGs

L4-L7 Service Graph Instances

Application profiles define the policies, services and relationships between EPGs. Each application profile can contain one or more EPGs that can communicate with the other EPGs in the same application profile and with EPGs in other application profiles according to the contract rules. Application network profiles are defined based on the communication, security and performance needs of the application. They are then used by the APIC to push the logical topology and policy definitions down to stateless network hardware in the fabric.

In the Cisco ACI design, the Cisco APIC provides automatic service insertion while acting as a central point of policy control and manages both the network fabric and services appliances. For service insertion, the APIC uses an L4-L7 service graph instance, which is essentially an ordered set of service function nodes between a set of terminals and a set of network service functions that are required by an application. The service graph instance can be defined by using GUI, CLI or the APIC. Multiple function nodes can be used in a service chain while creating an L4-L7 service graph. The APIC is responsible for pushing the needed configuration and security policies to the network switches, routers, firewalls, load balancers and other infrastructure components to create the data forwarding path required to force data traffic traversal through the service nodes as per the service graph requirements.

In Figure 11, the L4-L7 service graph example inserts the Cisco ASA Firewall and A10 Thunder ADC function nodes together in a service chaining configuration to provide firewall capabilities and server load balancing capabilities respectively between the consumer and provider EPGs. This example from APIC GUI configuration shows how to create an L4-L7 service graph template and insert various service nodes in the data forwarding path between the two EPGs. The APIC performs service graph rendering once the L4-L7 service graph template is applied to a security context, and it programs the network infrastructure including switches, firewall and load balancers.

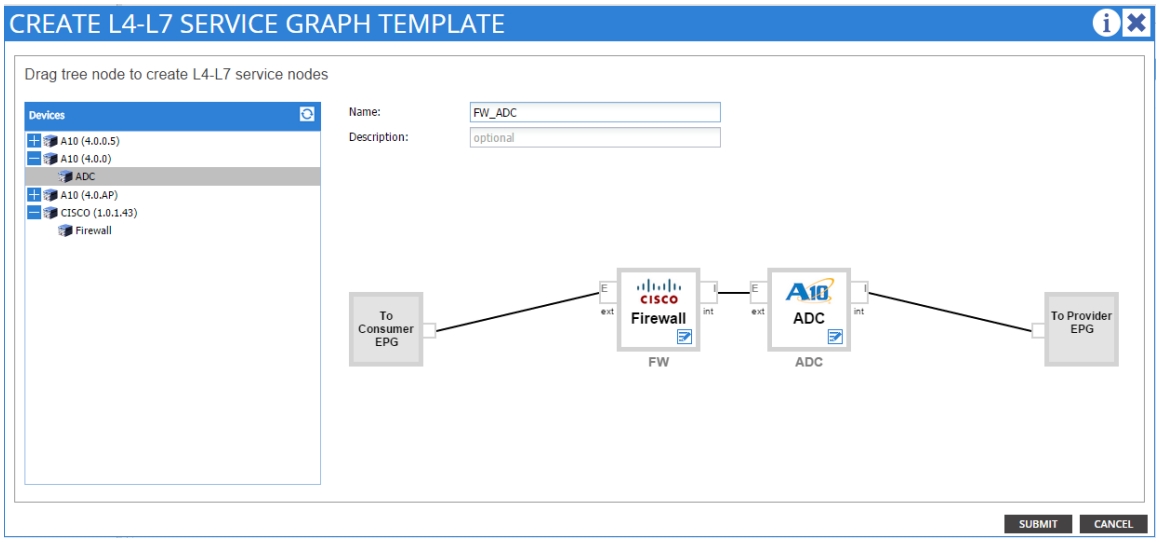

Figure 11: Service graph creation example

In the section below, we will refer to the simple two-tier application network profile shown in Figure 9 earlier and go through the various steps of creating an L4-L7 service graph template to connect the web EPG with the application EPG. Figure 12 shows how an APIC uses the service graph template to render a two-tier application profile onto the ACI fabric. The process involves interaction with the Nexus 9000 switches and the available pool of A10 Thunder ADC appliances to insert Thunder ADC server load balancing service between the application EPG and the web EPG.

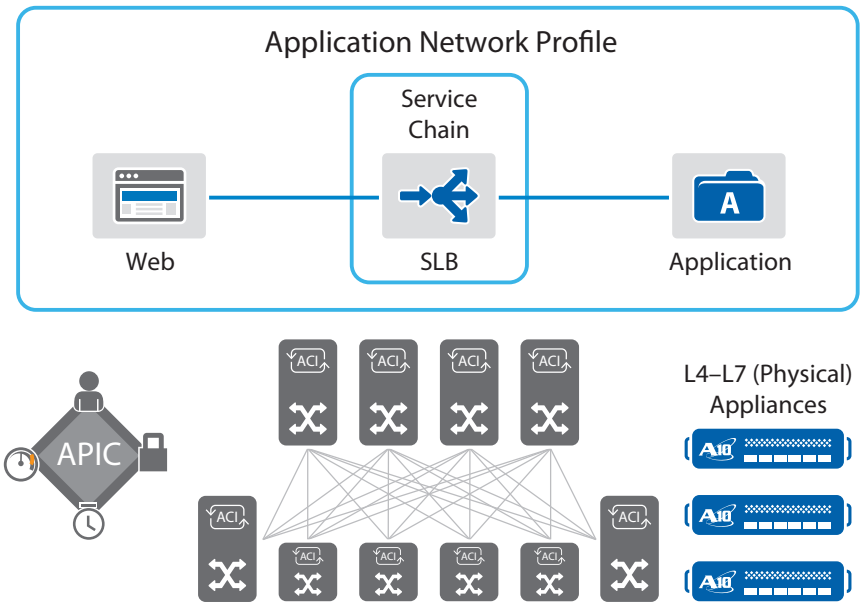

Figure 12: APIC rendering service graph to deploy application profile

Figure 13 shows a service graph template with an A10 Thunder ADC device between the consumer EPG and provider EPG. The web EPG is the consumer EPG in this case, and the application EPG is the provider EPG. The A10 Thunder ADC device provides server load balancing or application delivery controller services in this context.

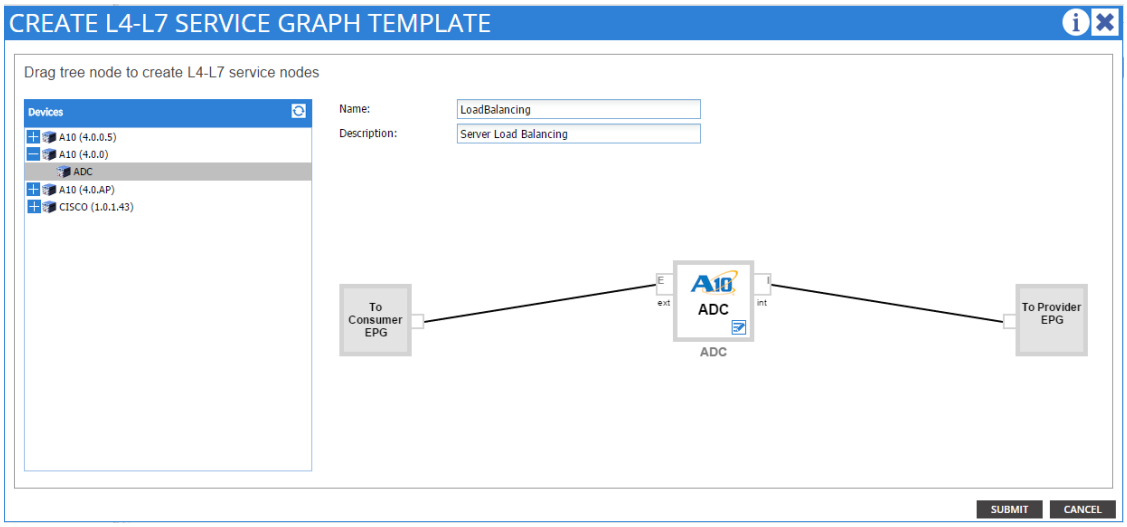

Figure 13: Service graph instance example

Cisco APIC configures the A10 Thunder ADC device once the service graph is applied to the security contract and also configures the data forwarding path. The A10 device to which the APIC pushes the configuration is selected through the L4-L7 device selection policy.

The service graph template shown in Figure 12 can be configured on the APIC in multiple ways as mentioned earlier in this document. The preferred method of configuring service graphs on the APIC is through XML payloads sent to the APIC using REST API calls.

The A10 device package enables Cisco APIC to configure ADC server load balancing configuration parameters defined in the service graph configuration. In this example, the service graph allows the user to configure the following parameters for A10 Thunder ADC load balancing service:

- Multiple real servers with IP address, port number, etc.
- Service-group to pool multiple real servers into a group
- Virtual server IP (VIP) for application services
- Health monitoring rules to monitor the state of real servers
- Network Address Translation (NAT) pools to perform source and destination address translation

The service graph itself can be configured either through the APIC GUI or can be configured through an XML post to the APIC. The service graph configuration requires some parameters that are marked as mandatory, for example the virtual server name, IP address and port number. The user can configure one or more service graph templates under the tenant; however, the service graph renderings are done only when the graph is applied to a security contract as shown in Figure 15. When the service graph is rendered, the APIC communicates with the A10 device using aXAPI commands and configures interfaces, VLANs and server load balancing specific configuration parameters. The A10 device package makes the distinction between a physical A10 device and virtual A10 device so that the Ethernet interface on the A10 device can be configured as a tagged or untagged port. Figure 14 shows the ADC configuration in the APIC once the graph has been associated with a contract.

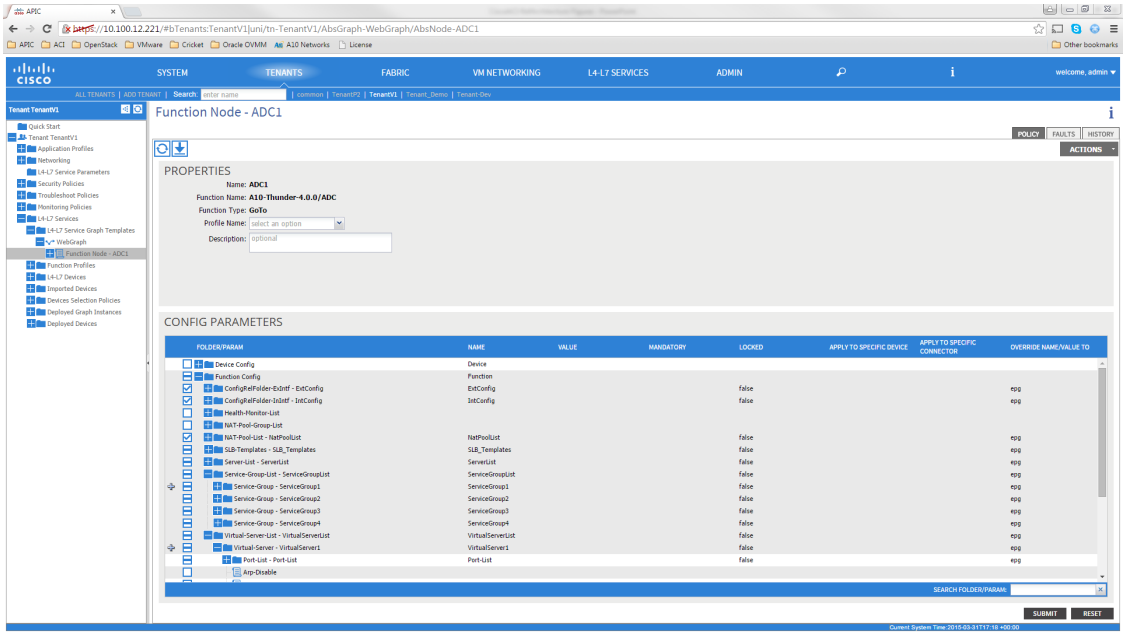

Figure 14: Service graph showing L4-L7 function node configuration

Figure 15 shows how the graph is associated with a contract. EPGs can only communicate with other EPGs as per the contract rules, and inter-EPG communication is disabled by default in ACI. The function node gets inserted in inter-EPG communication through the service graph.

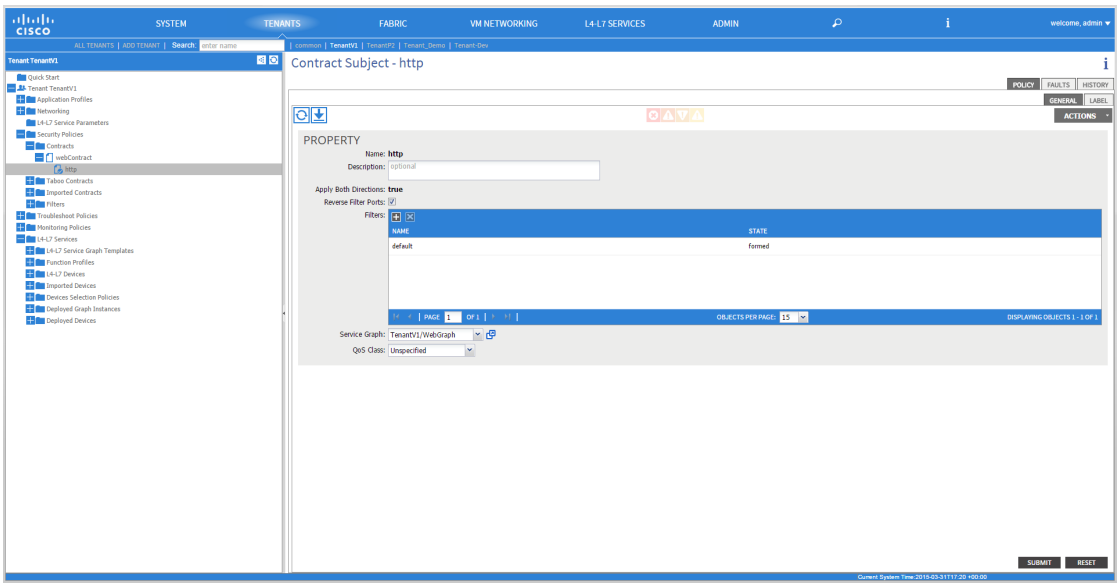

Figure 15: Associating service graph with a security contract

Once the service graph is applied and rendered, the A10 device will receive configuration through REST API commands from the APIC. Table 1 shows the A10 Thunder ADC load balancing configuration after APIC has rendered the service graph and pushed configuration to it. The management interface configuration is preconfigured and doesn't change. The Ethernet interface configuration, servers, service group and virtual servers are configured by the APIC controller. The A10 Thunder ADC device configuration is managed and updated by APIC in this deployment model and doesn't require any manual interaction with the Thunder ADC devices.

The service graph status can be checked in the APIC GUI; if the service graph is rendered successfully, then the status code is "applied," but if the service graph rendering fails due to some error, the status code is "failed-to-apply." Figure 16 shows the service graph rendering status on the APIC GUI.

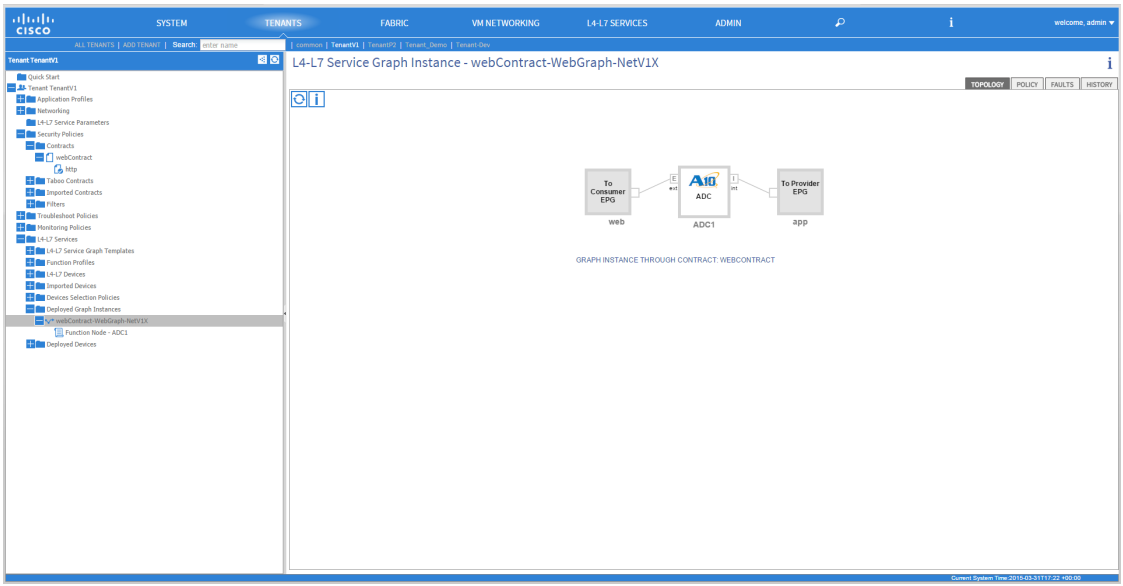

Figure 16: APIC GUI showing Deployed Service Graph Instances

Adding L4-L7 Devices to the APIC

The L4-L7 devices can be added as either physical appliances attached to the leaf switches or they can be virtual appliances running on virtualized servers. Cisco ACI treats the L4-L7 devices like load balancers, and firewalls as L4-L7 concrete devices with concrete interfaces. However, the L4-L7 service graph is composed of abstract nodes. The L4-L7 concrete devices have to be configured in a device cluster configuration and this abstraction is called a logical device. The logical device has logical interfaces that are mapped to concrete interfaces on the concrete device.

The device selection policy provides a device cluster or logical device that can be selected based on a contract name, a graph name, or the function node name inside the service graph. Device selection policy is depicted in Figure 17. The device selection policy dictates the device cluster used during service graph rendering. The APIC allocates VLAN for a function node connector during service graph rendering to allocate VLAN to the logical interfaces of the device cluster. VLAN information is pushed to the concrete interfaces during service graph instantiation.

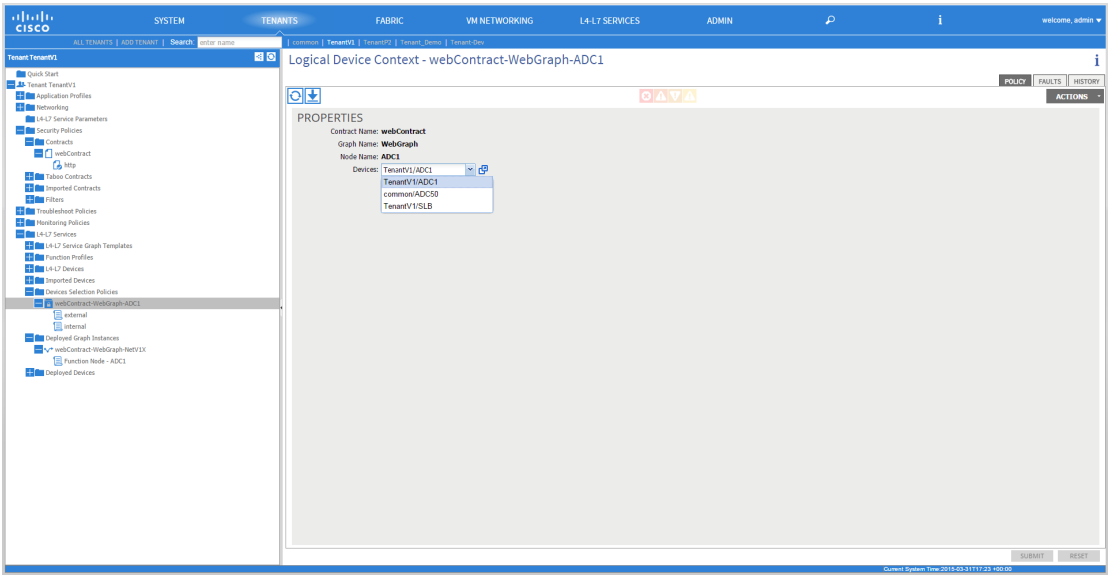

Figure 17: L4-L7 device selection policy

Figure 18 shows how to add an L4-L7 device (concrete and logical) from the APIC GUI. The physical interfaces Ethernet 11 and Ethernet 12 that are attached to the leaf switches 105 and 106 can be specified here. The APIC also allows you to specify the direction for the attached interfaces.

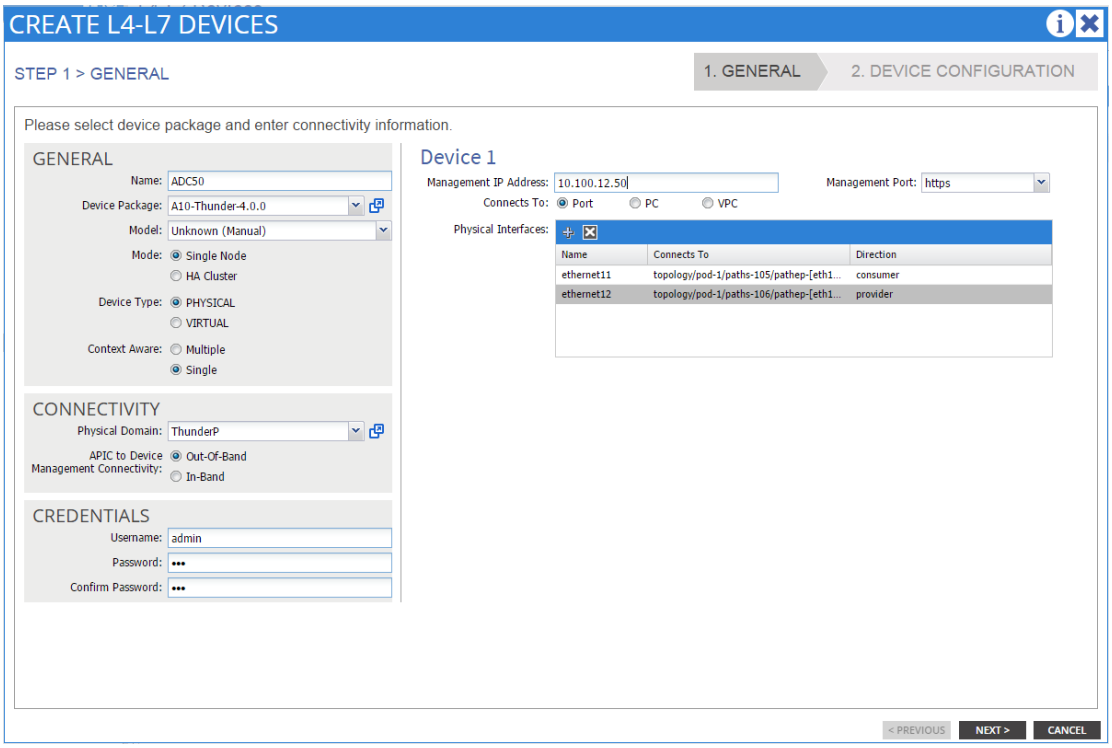

Figure 18: Adding L4-L7 device to tenant

Figure 19 shows the L4-L7 device status (concrete and logical) from the APIC GUI. The cluster configuration is complete and the logical device has the logical interfaces named external and internal (Lif) mapped to the concrete device interfaces (Cif) e11 and e12. During service graph rendering, APIC pushes the configuration to the concrete device with the VLAN information for the concrete interfaces based on the logical interface to concrete interface mapping.

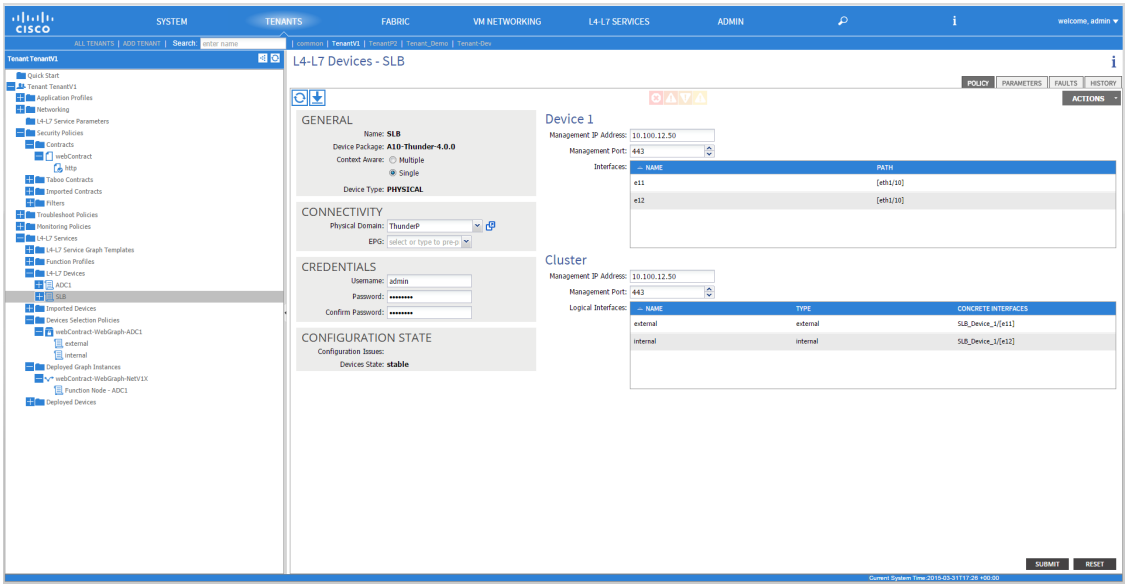

Figure 19: L4-L7 device status in APIC GUI

Figure 20 shows the logical device context configuration from APIC GUI. This configuration helps the APIC choose the logical device for the given contract, service graph and node during the service rendering step.

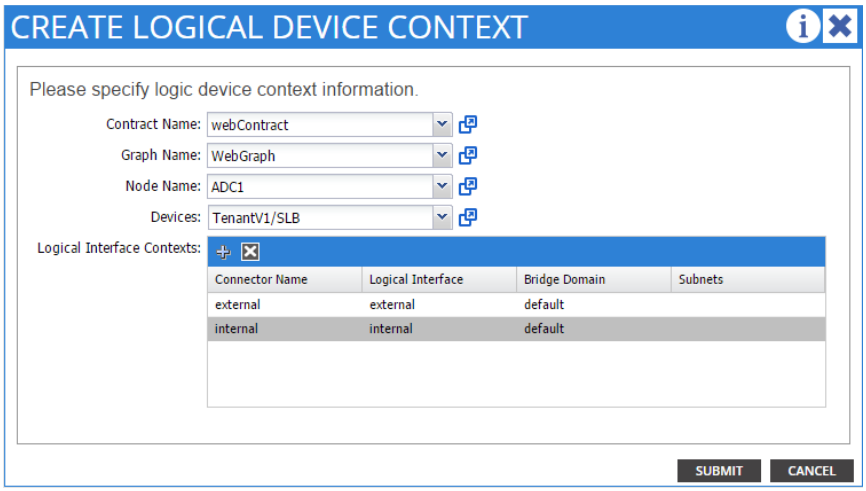

Figure 20: Creating a logical device context

Verifying L4-L7 Service Graph Configuration

Figure 21 shows the L4-L7 deployed graph status with the service graph configuration information that shows the VLAN allocated to the connector interfaces.

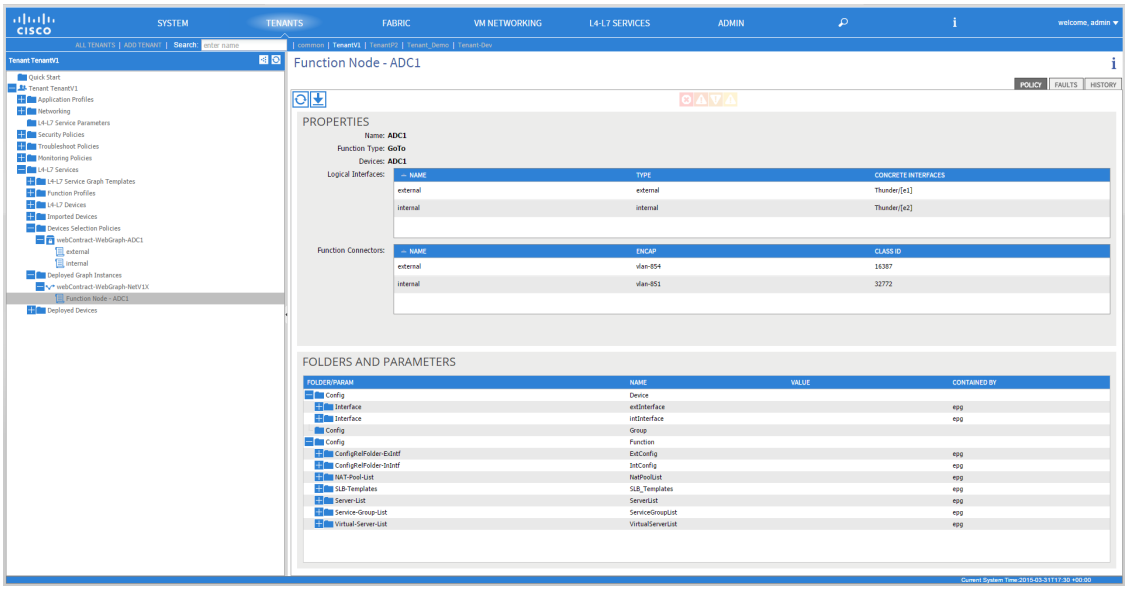

Figure 21: APIC GUI deployed graph configuration information

The A10 Thunder ADC device configuration is shown in the code below.

```
vlan 819
  untagged ethernet 2
  router-interface ve 819
  name internal
!
vlan 820
  untagged ethernet 1
  router-interface ve 820
  name external
!
! pre-existing configuration
interface management
  ip address 10.100.12.91 255.255.255.0
  ip default-gateway 10.100.12.1
!
!
interface ethernet 1
  enable
!
interface ethernet 2
  enable
!
!
interface ve 819
  name intInterface
  ip address 20.20.20.242 255.255.255.0
!
interface ve 820
  name extInterface
  ip address 10.10.10.242 255.255.255.0
!
!
ip nat pool Pool1 10.10.10.151 10.10.10.152 netmask /24
!
ip nat pool Pool2 10.10.10.161 10.10.10.162 netmask /24
!
!
slb template tcp TCP_1
  idle-timeout 3000
!
!
slb template virtual-server VS_1
  conn-limit 100 reset no-logging
  conn-rate-limit 50
!
!
slb server S1 20.20.20.51
  port 21 tcp
  port 22 tcp
  port 53 udp
  port 80 tcp
  port 8080 tcp
!
slb server S2 20.20.20.52
  port 21 tcp
```

```
  port 22 tcp
  port 53 udp
  port 80 tcp
  port 8080 tcp
!
slb server S3 20.20.20.53
  port 21 tcp
  port 22 tcp
  port 53 udp
  port 80 tcp
  port 8080 tcp
!
!
slb service-group sg-tcp21-1 tcp
  member S1 21
  member S2 21
    priority 10
  member S3 21
    priority 15
!
slb service-group sg-tcp22-1 tcp
  member S1 22
  member S2 22
    priority 10
  member S3 22
    priority 15
!
slb service-group sg-tcp80-1 tcp
  member S1 80
  member S2 80
    priority 10
  member S3 80
    priority 15
!
slb service-group sg-udp53-1 udp
  member S1 53
  member S2 53
    priority 10
  member S3 53
    priority 15
!
!
slb virtual-server VIP1 10.10.10.101
  port 22 tcp
    service-group sg-tcp22-1
  port 53 udp
    service-group sg-udp53-1
  port 80 tcp
    source-nat pool Pool1
    service-group sg-tcp80-1
    template tcp TCP_1
  port 8080 http
    source-nat pool Pool2
    service-group sg-tcp80-1
  port 8081 fast-http
    service-group sg-tcp80-1
```

Reference Architecture Components

Cisco Nexus 9000 Series Switches

- 2 x N9K-C9396PX running n9000-11.0(2m) deployed as leaf switches
- 2 x N9K-C9336PQ running n9000-11.0(2m) deployed as spine switches

Cisco APIC Cluster

- 3 x APIC servers running on UCS server running apic-1.0(2m)

A10 Thunder ADC appliances

- A10 Thunder 3030S ADC appliance running ACOS 4.0.0
- A10 vThunder ADC line of virtual appliances running on VMware ESXi Server running ACOS 4.0.0

A10 device package

- A10 device package version 4.0.0 ((minor version 1.0(1.0))

VMware vCenter

- VMware vCenter running vSphere 5.5.1

VMware ESXi servers

- VMware ESXi version 5.5.1

Summary

Cisco ACI provides centralized, application-driven policy automation and management at all layers of the networking infrastructure. A10 Thunder ADC integrates with the Cisco ACI open ecosystem model and provides dynamic L4-L7 application networking services that can be automated and managed by Cisco Application Policy Infrastructure Controller (APIC). The result is business agility and lower IT costs by increasing application scalability, reducing the complexity and the time it takes to deploy new applications. The integrated solution can be leveraged by customers to achieve agile application rollout and meet SLAs and security requirements consistently.

Cisco APIC can dynamically insert rich application delivery and security capabilities of A10 Thunder ADC into a shared multi-tenancy infrastructure with the A10 device package. This joint solution can effectively transform the data center as applications and services are created with significant increase in provisioning speed, business agility and reduced costs.

About A10 Networks

A10 Networks is a leader in application networking, providing a range of high-performance application networking solutions that help organizations ensure that their data center applications and networks remain highly available, accelerated and secure. Founded in 2004, A10 Networks is based in San Jose, California, and serves customers globally with offices worldwide. For more information, visit: **www.a10networks.com**

Corporate Headquarters

A10 Networks, Inc
3 West Plumeria Ave.
San Jose, CA 95134 USA
Tel: +1 408 325-8668
Fax: +1 408 325-8666
www.a10networks.com

Part Number: A10-RA-30100-EN-01
Apr 2015

Worldwide Offices

North America
sales@a10networks.com
Europe
emea_sales@a10networks.com
South America
latam_sales@a10networks.com
Japan
jinfo@a10networks.com
China
china_sales@a10networks.com
Taiwan
taiwan@a10networks.com
Korea
korea@a10networks.com
Hong Kong
HongKong@a10networks.com
South Asia
SouthAsia@a10networks.com
Australia/New Zealand
anz_sales@a10networks.com

To learn more about the A10 Thunder Application Service Gateways and how it can enhance your business, contact A10 Networks at: **www.a10networks.com/contact** or call to talk to an A10 sales representative.

©2015 A10 Networks, Inc. All rights reserved. The A10 logo, A10 Harmony, A10 Lightning, A10 Networks, A10 Thunder, aCloud, ACOS, ACOS Policy Engine, Affinity, aFleX, aFlow, aGalaxy, aVCS, AX, aXAPI, IDaccess, IDsentrie, IP-to-ID, SoftAX, SSL Insight, Thunder, Thunder TPS, UASG, VirtualN, and vThunder are trademarks or registered trademarks of A10 Networks, Inc. All other trademarks are property of their respective owners. A10 Networks assumes no responsibility for any inaccuracies in this document. A10 Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.